공학석사 학위논문

신조어 및 띄어쓰기 오류에 강인한 시퀀스-투-시퀀스 (Sequence-to-sequence) 기반 한국어 형태소 분석기

Sequence-to-sequence based Korean Morphological Analyzer for Neologism and Spacing Error

2019 년 8 월

서울대학교 대학원

컴퓨터공학부

최 병 서

신조어 및 띄어쓰기 오류에 강인한 시퀀스-투-시퀀스 (Sequence-to-sequence) 기반 한국어 형태소 분석기

Sequence-to-sequence based Korean Morphological Analyzer for Neologism and Spacing Error

지도 교수 이 상 구

이 논문을 공학석사 학위논문으로 제출함

2019 년 6 월

서울대학교 대학원

컴퓨터공학부

최 병 서

최병서의 공학석사 학위논문을 인준함

2019 년 6 월

위 원 장 이 재 진 (인)

부위원장 이 상 구 (인)

위 원 이 창 건 (인)

초 록

최근 인터넷 텍스트 데이터의 수가 늘어나면서 인터넷 텍스트의 자연어 처리에 대한 수요도 늘고 있다. 특히, 한국어 커뮤니티 등에서 수집되는 한국어로 된 텍스트 데이터를 처리해야 할 필요성이 늘고 있다. 그런데 한국어의 교착어라는 특성 상 한국어 자연어 처리에는 형태소 분석이 필수적이라 할 수 있다.

특히 인터넷 텍스트 데이터를 분석하기 위해서는, 띄어쓰기 오류가 있는 문장에서도 정확히 형태소 분석을 해내야 하고, 신조어 등의 OOV 입력에 대한 원형 복원 성능이 충분해야 한다. 그러나 기존 한국어 형태소 분석기는 원형 복원에 사전을 사용하는 경우가 많고, 이를 위해 기분석 사전 또는 규칙에 기반한 전처리 과정 등이 필요하다.

본 논문에서는 Sequence-to-sequence 모델을 기반으로 위의 띄어쓰기 문제와 신조어 문제를 효과적으로 처리할 수 있어 인터넷 텍스트 데이터 분석을 효과적으로 할 수 있는 한국어 형태소 분석기 모델을 제안하였다. 본 모델은 높은 확장성을 위하여 사전을 사용하지 않고, 규칙에 기반한 전처리를 최소화하였다. 또한 본 논문에서 제안하는 모델은 일반적으로 사용하는 음절 외에도 음절 Bigram과 자소라는 두 가지 추가 요소를 입력 자질로 같이 사용하였다. 그리고 어절 구분 정보에 의존하지 않도록 공백을 제거한 데이터를 학습 데이터로 같이 사용하였다.

실험 결과 세종 말뭉치에서 형태소 단위 F1-measure가 0.9793으로, 기존 형태소 분석기와 비교해서 부족하지 않고 사전을 사용하지 않는 다른 형태소 분석기에 비해 뛰어난 성능이 나왔다. 띄어쓰기가 없는 데이터셋에 대해서도 1% 전후의 성능 감소만이 있었으며, Train 데이터셋에 없는 단어 및 인터넷 데이터 샘플에 대해서도 높은 성능이 나오는 것을 확인하였다.

주요어 : 형태소 분석, 품사 태깅, 시퀀스 투 시퀀스, 원형 복원, 인터넷 텍스트 데이터

학 번 : 2017-25969

목 차

표 목차

그림 목차

제 1 장 서론

1.1 연구의 배경 및 내용

자연어 처리는 인간의 언어, 즉 자연어로 된 데이터를 컴퓨터를 이용하여 분석하는 인공지능의 한 분야이다. 특히 소셜 네트워크 서비스(SNS), 웹 문서를 비롯한 인터넷 텍스트 데이터는 기하급수적으로 증가하고 있고, 이 빅 텍스트 데이터를 정보 검색, 번역, 챗봇, QA 시스템, 키워드 추출 등 다양한 분야에 활용하려는 시도가 이어지고 있다.

그런데 빅 텍스트 데이터, 특히 한국어 빅 텍스트 데이터를 활용하기 위해서는 형태소 분석이 필수적이다. 글로 된 자연어 문장의 의미를 이해하고 처리하기 위한 첫 번째 단계는 각 문장을 단어 또는 형태소와 같이 의미를 가지는 기본적인 단위로 나누는 것이다. 굴절어에 해당하는 영어의 경우, 대부분의 단어가 공백으로 구분되고, 굴절 역시 제한적이다. 따라서 규칙에 기반한 간단한 전처리만으로 단어의 원형을 복구하여 의미 분석의 기본 단위로 사용할 수 있다. 하지만 교착어인 한국어는 발음의 단위인 어절로 띄어쓰기가 되어있고, 한 어절은 의미를 가진 최소 단위인 형태소 여러 개의 결합으로 이루어져 있어 공백만을 기준으로 의미 경계를 나누기엔 부족하다. 또한 그 형태소의 결합 과정에서 활용 등으로 인해 원래 형태소의 형태가 변화하는 현상이 발생한다. 따라서 한국어 문장을 분석하기 위해서는 어절 단위로 나뉜 문장을 형태소 단위로 나누고, 형태소의 원래 형태로 복원하는 형태소 분석이 필요하다. 따라서 형태소 분석은 한국어 자연어 처리를 위해 필요한 가장 기초적인 과정이라고 할 수 있다.

그런데 최근 많이 늘어나고 있는 인터넷 빅 텍스트 데이터를 분석하기 위해서는 여러 가지를 추가적으로 고려해야 한다. 인터넷

데이터의 문장은 일반적인 문장에 비해 문법 오류가 많은데, 특히 띄어쓰기가 잘못된 문장이 많다. 따라서 인터넷 데이터를 분석하기 위한 형태소 분석기는 띄어쓰기 오류에 강건하여 어절이 올바르게 분리되어 있지 않는 문장도 올바르게 분석할 수 있어야 한다.

또한 인터넷 데이터에는 기존에 존재하지 않는 신조어 및 고유명사가 많이 나타난다. 이런 어휘 외 단어, 즉 OOV(Out of vocabulary)에서 오분석이 발생할 경우 형태소 분석 이후로 이어지는 작업들에 그 오류가 누적되게 된다. 따라서 OOV 입력에 대한 잘못된 분석을 최소화하고 원형을 정확히 추출해내는 형태소 분석기가 필요하다.

그러나 기존의 많은 형태소 분석기는 원형 복원 과정 등에서 사전을 이용한다. 이 경우 학습 데이터셋에서 입력 어절과 출력 형태소를 대응시켜 사전을 구축하는 전처리 과정이 필요하고, 학습 데이터셋에는 나타나지 않은 형태의 형태소 결합이 나타날 경우 원형 분석 사전을 이용해서 복원하는 것에는 한계가 있다. 또한 문맥에 따라 같은 표현형에서 다른 원형 복원이 이루어지는 경우 이를 분석하기 어렵다.

따라서 본 논문에서는 위의 어절 분리 문제와 OOV 문제를 효과적으로 처리할 수 있어 인터넷 텍스트 데이터 분석을 효과적으로 할 수 있는 형태소 분석기 모델을 제안한다. 본 논문에서 제안하는 모델은 시퀀스 투 시퀀스(sequence-to-sequence)를 이용하는 모델로, 원형 복원 사전을 비롯한 언어 지식을 사용하거나 규칙 기반의 전처리 과정을 거치지 않고 입력 문장에서 End-to-end로 원형 복원까지 형태소 분석을 한다. 또한 입력 문장에 공백이 제대로 되어 있지 않은 경우에도 효과적으로 대응할 수 있도록 어절 경계에 의존하지 않는 방법으로 학습을 한다. 그리고 음절 bigram과 자소 임베딩을 추가 입력 자질로 사용하여 형태소 분석 성능을 높인다.

본 논문에서는 제안하는 모델의 형태소 분석 성능을 확인하기 위해 일반적으로 사용하는 정제된 한글 말뭉치인 세종 말뭉치에 대한 성능을

측정하여 기존에 연구된 형태소 분석기 모델과 비교하여 본 모델이 기존 모델에 비해 경쟁력이 있고, 특히 사전을 사용하지 않는 모델들보다 성능이 높음을 확인한다. 또한 인터넷 데이터에서의 성능을 확인하기 위해 띄어쓰기가 제거된 데이터 및 인터넷에서 직접 수집한 텍스트 데이터에 대한 형태소 분석 성능을 기존 공개 형태소 분석기와 비교하여 본 모델이 띄어쓰기가 잘못된 입력에 대해서도 충분한 성능을 보장함을 확인한다. 그리고 학습 데이터에 없거나 적게 나타나는 형태소에 대한 분석 성능을 비교하여 본 모델이 고유명사나 인터넷 신조어를 잘 분석할 수 있음을 확인한다.

1.2 논문의 구성

본 논문의 구성은 다음과 같다. 2장에서는 보편적인 형태소 분석기의 분석 과정을 소개하고, 기존 한국어 형태소 분석기의 형태소 분석 방법에 대해 살펴본다. 3장에서는 본 논문에서 제안하는 형태소 분석 모델을 설명한다. 4장에서는 기존 형태소 분석기와 본 논문에서 제안한 형태소 분석기 모델의 형태소 분석 성능을 비교하고, 특히 등장 빈도수가 낮은 형태소에 대한 분석 성능을 비교한다. 마지막으로 5장에서는 본 논문의 결론과 향후 연구 방향에 대하여 논한다.

제 2 장 관련 연구

이 장에서는 지금까지의 한국어 형태소 분석기 관련 연구를 소개한다. 1절에서는 일반적인 한국어 형태소 분석기의 형태소 분석 과정을 3단계로 나눠서 설명하고, 2절에서는 기존에 어떤 한국어 형태소 분석 연구가 있었는지 살펴본다.

2.1 한국어 형태소 분석 과정

한국어에서 형태소 분석(Morphological analysis)라고 하면 어절 단위로 나뉜 한국어 문장을 형태소로 나누고, 형태소의 원래 형태를 복원하고, 각 형태소에 품사를 다는 일련의 과정을 말한다[7]. 이 세 과정을 각각 형태소 분할(Morpheme segmentation), 원형 복원(Original form recovery), 품사 태깅(Part-of-speech tagging)이라고 한다. 형태소 분석기마다 세부적인 순서 등은 다르지만 최근의 연구는 주로 이 세 가지 과정을 포함하고 있다. 표 1에 형태소 분석의 세 단계 과정의 예시를 나타냈다.

표 1 한국어 형태소 분석 과정의 예시

원본 어절	굴렀다		
형태소 분할	굴렀		다
품사 태깅	굴렀/VV+EP		다/EF
원형 복원	구르/VV	었/EP	다/EF

2.1.1 형태소 분할

형태소 분석을 위해서는 기본적으로 하나 이상의 형태소 조합으로 이루어진 어절 안에서 형태소를 나누는 작업이 필요하다. 확률 기반 모델에서는 사전을 이용하여 음절 또는 자소 단위로 어절을 나눠 그 중 가능한 형태소 시퀀스 후보를 모두 고려하는 방식을 사용하였다. [7],

[13]과 같은 딥러닝을 이용한 연구들에서는 주로 음절 단위로 형태소 분할을 하며, 품사 태깅과 통합하여 진행되는 연구 또한 많다.

2.1.2 품사 태깅

품사 태깅은 형태소 분할 과정을 거친 형태소에 품사(Part-of-speech Tag)를 부착(Tagging)하는 과정이다.

형태소의 품사는 형태소를 의미나 형식 등에 따라 명사, 형용사 등으로 분류한 것이다. 형태소의 품사 분류 기준은 여러 가지가 있지만, 일반적으로 한국어 형태소 분석 연구에서는 세종 말뭉치에서 사용하는 42개 품사를 기준으로 사용한다.

엄밀하게는 형태소 분석과 품사 태깅은 조금 다른 과정이지만 본 논문을 포함하여 일반적으로 한국어 형태소 분석은 품사 태깅 과정을 포함해서 말한다. 확률 기반 형태소 분석 모델에서는 가능한 형태소 분할 후보 중 가장 적합한 형태소 시퀀스를 선택하는 데 품사 정보를 사용하기 때문에 형태소의 형태와 그 품사가 같이 결과로 나오고 형태소 분석 결과는 자연스럽게 품사 부착 결과를 포함한다. 또한 확률 기반 모델이 아닌 경우에도 품사 정보를 형태소 원형 복원 과정 등에 사용하는 모델이 많았다. 또한 형태소 분할 및 원형 복원에 품사 정보를 사용하지 않는 모델이다 하더라도 형태가 같은 형태소에 대해 품사 정보는 중의성을 해소해 줄 수 있는 추가적인 정보가 되고, 형태소 분석에 이어지는 구문 분석 등의 과정에 활용될 수 있다. 따라서 대부분의 한국어 형태소 분석 연구는 품사 태깅 과정을 포함하며, 형태소 분석 결과로 품사가 부착된 형태소 시퀀스를 출력한다.

2.1.3 원형 복원

원형 복원은 표현형(surface form)으로 되어 있는 형태소를 원형(original form)으로 복원하는 과정이다. 한국어의 특성 상 단순히 글자를 나누는 것만으로는 원래 형태소의 형태를 얻어낼 수 없다. 예를 들어 한국어의 용언은 활용이 일어나는데, 이 활용 과정에서 어간과 어미의 결합 과정에서 음절의 수나 형태가 변하게 된다. 용언의 원형인 어간을 복원할 수 없다. 따라서 문장을 형성하면서 변화된 모습의 형태소, 즉 형태소의 표현형을 형태소 결합 전의 원형으로 복원하는 작업이 필요하다.

기존 한국어 형태소 분석기 모델의 원형 복원은 주로 말뭉치를 통해 구축한 원형 복원 사전을 통해 이루어져왔다[14][16]. 일반적으로 말뭉치에서 음절 간 대응 관계를 정렬(align)하여 그림 1과 같이 원형 사전을 구축한다. 그리고 원형 복원 단계에서는 원형 복원이 필요한 형태소에 대해 사전을 통해 원형을 복원한다.

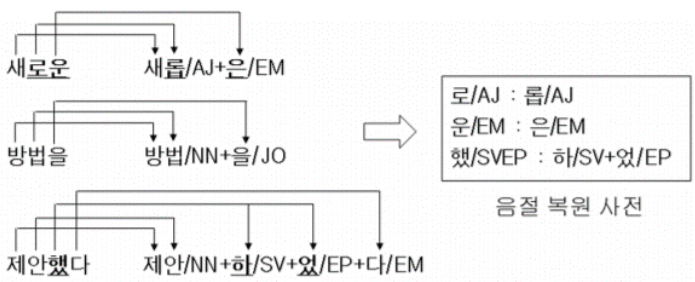

그림 1 음절 원형 복원 사전 생성 과정 예시[16]

하지만 사전을 사용하는 원형 복원의 경우 몇 가지 한계가 존재한다. 첫째로 학습 데이터셋에는 나타나지 않은 형태의 형태소 결합이 나타날 경우 원형 분석 사전을 이용해서 복원하는 것에는 한계가 있다. 예를 들면, 조사 '는', '를'의 축약형 'ㄴ', 'ㄹ'과 같은 경우는 받침이 없는

모든 단어와 결합할 수 있다. 원형 복원 사전이 형태소나 어절 단위로 구성될 경우 모든 단어에 대한 이와 같은 복원을 담을 수는 없다.

또한 사전을 이용한 원형 복원은 문맥에 따른 중의성을 해소하는 것이 쉽지 않다. 예를 들어, “소리를 들었다”의 “들었다”와 “물건을 들었다”의 “들었다”의 경우 원형 복원을 제외하면 똑같이 “들/VV + 었/EP + 다/EF”로 형태소 분석되지만, 의미를 따져보면 “소리를 들었다”의 동사의 원형은 “듣다”이고, “물건을 들었다”의 동사의 원형은 “들다”이기 때문에 원형 분석 결과는 각각 “듣/VV”과 “들/VV”로 달라져야 한다. 하지만 원형 복원 사전을 적용하는 방식의 경우 이런 모호성을 해소하는 것이 굉장히 어렵다.

[1], [7]등의 연구에서는 원형 복원을 품사 태깅과 같이 시퀀스 레이블링(Sequence Labeling) 문제로 접근하여 사전을 사용하지 않고 원형 복원을 하는 방법을 연구하였다. 하지만 연속적 레이블링에 기반한 방식은 사전을 사용한 방식과 마찬가지로 음절의 대응관계를 맞춰줘야 하는 정렬 전처리가 필요하다는 문제가 있다.

2.2 기존 한국어 형태소 분석기 연구

전통적으로 형태소 분석기는 규칙과 은닉 마르코프 모델(Hidden Markov Model, HMM)에 기반한 형태소 분석기[5][18] 등의 방법이 주로 연구되었다. kkma[①] 형태소 분석기는 인접 조건 검사[15]를 기본으로, 은닉 마르코프 모델 기반한 확률 모델(Probabilistic Model)과 여러 규칙을 같이 이용한다. Komoran[②]은 자소 단위로 TRIE 사전 탐색을 통해 형태소 후보를 얻어낸 후 lattice 격자를 은닉 마르코프 모델을 통해 분석한다.

---

① http://kkma.snu.ac.kr/
② https://www.shineware.co.kr/products/komoran/

[8], [9] 등은 CRF(Conditional random field)를 이용하여 형태소 분할과 품사 태깅을 한 뒤, 사전 및 은닉 마르코프 모델을 기반으로 원형 복원을 하였다.

최근에는 딥 러닝을 통한 한국어 형태소 분석기 연구 또한 활발하게 이루어지고 있다. 딥 러닝을 이용한 한국어 형태소 분석기는 음절 단위의 품사 태깅 방법론을 활용한 연속적 레이블링(Sequence labeling) 문제로 많이 연구되고 있다[13]. 이를 위해 순환 신경망(Recurrent neural net, RNN)을 이용하는 방식이 주로 연구되었다.

[13]은 BiLSTM-CRF을 사용하여 음절 단위 형태소 분할과 품사 복원을 동시에 진행 후 여러 사전을 순차적으로 사용하여 원형 복원을 하였다. 이 연구에서는 N-gram이나 명사 사전 등을 다양한 요소를 입력으로 사용하여 높은 성능을 냈다.

Khaiii [③] 는 주로 사용되는 순환 신경망 대신 컨볼루션 신경망(Convolutional Neural Network, CNN)을 사용하여 음절 단위로 형태소 분석을 하였다.

[1], [7] 등은 BiLSTM 또는 BiLSTM-CRF 모델을 사용하여 원형 복원 사전을 사용하지 않는 형태소 분석기를 연구하였다. [1]은 자소 단위로 BiLSTM을 이용하여 형태소 원형을 먼저 복원한 뒤 BiLSTM-CRF를 이용하여 형태소 분할 및 품사 태깅을 하였다. [7]은 BiLSTM을 사용하여 형태소 분할 및 원형 복원을 한 뒤, 또 하나의 BiLSTM을 사용하여 품사 태깅을 하였다.

한편, 시퀀스 투 시퀀스(Sequence-to-sequence)[12]를 사용한 End-to-end 방식의 형태소 분석기 연구 역시 이루어지고 있다. 시퀀스 투 시퀀스 방식의 형태소 분석기는 형태소 분석을 일종의 번역 문제로

[③] https://github.com/kakao/khaiii

본다. 이 방식은 기본적으로 입출력 시퀀스의 길이 및 형태가 같을 필요가 없기 때문에 형태소 분할, 원형 복원 및 품사 태깅의 과정을 한 번에 처리할 수 있다. [17]는 시퀀스 투 시퀀스에 합성곱 요소(convolutional feature)를 사용하는 시도를 하였다. [3]은 기존 시퀀스 투 시퀀스 모델에 입력 추가 구조(input-feeding)과 복사 방법론(copying mechanism)을 적용하여 성능을 비교하였다.

제 3 장 모델 설명

이 장에서는 본 논문에서 제안하는 형태소 분석기 모델에 대해 설명한다. 1절에서는 모델 전체 구조 및 장점에 대해 설명하고, 2절에서는 모델의 입력 및 출력이 어떤 형태로 이루어지는지 설명하고, 3절에서는 형태소 분석 작업을 수행하는 주요 신경망인 주의 기반 인코더-디코더 모델 구조에 대해 구체적으로 설명한다.

3.1 모델 구조

본 논문의 모델은 기본적으로 음절 단위 시퀀스 투 시퀀스를 통해 형태소 분할과 품사 태깅, 원형 복원을 동시에 하는 End-to-end 방식 모델이다. 이 모델은 어휘사전 및 특수한 전처리가 필요 없고, 품사 태깅을 하지 않고도 키워드를 분리해낼 수 있다. 또한 기존 시퀀스 투 시퀀스 방식 연구와 달리 음절 Bigram과 자소 요소를 추가적으로 인코더의 입력으로 사용한다.

모델의 전체 구조는 그림 2과 같다. (1)에서 입력 문장을 음절 단위로 임베딩, (2)에서 인코딩하여 이를 바탕으로 (3)에서 형태소 분석된 문장을 음절 단위로 출력한다. 모델의 각 부분에 대해서는 2절과 3절에서 자세히 설명한다.

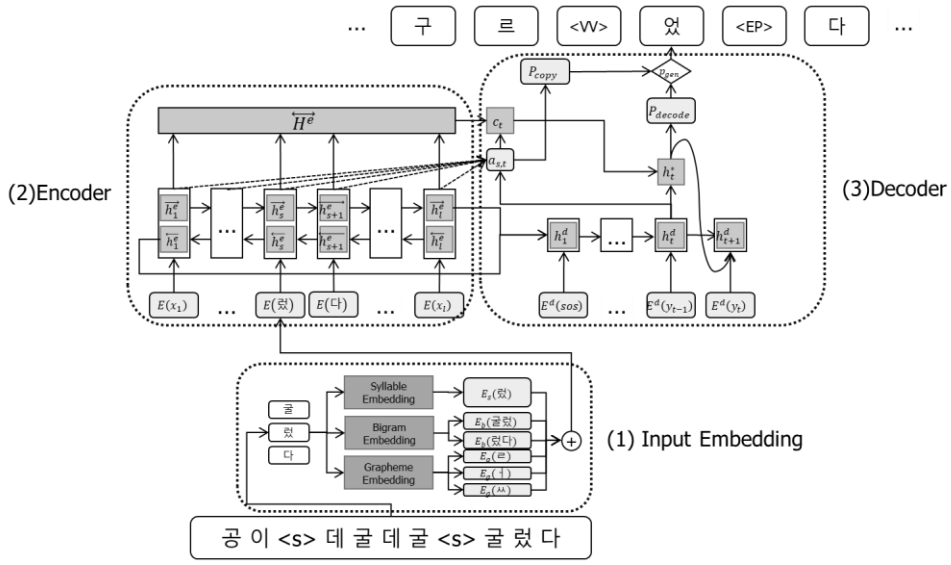

그림 2 본 논문의 형태소 분석기 모델 구조

3.2 입력 및 출력

3.2.1 입출력 단위 및 형태

본 모델에서는 한국어 음절 신경망의 기본 입력 및 출력 단위로 사용하였다. 따라서 입력 문장을 음절 단위로 나누어 각 음절을 임베딩한 벡터를 신경망의 입력으로 넣었다. 띄어쓰기가 입력에 있을 경우 띄어쓰기 태그 <s>로, 음절 입력과 동등한 입력으로 처리하였다. 한글 외의 문자가 입력으로 들어올 경우 그 문자를 음절과 동등하게 학습하였다.

출력은 음절과 품사 태그로 구성된다. End-to-end를 위해, 디코더에서는 음절의 시퀀스와 품사 태그가 반복되어 출력된다. 음절이 연속적으로 출력되는 경우 이 음절들은 1개의 형태소로 보고, 그 형태소의 품사는 연속되는 출력 끝에 등장한 품사 태그가 태깅된다. 이렇게 되면 품사 태그가 형태소 간 구분자 역할을 겸한다. 출력에서는

어절 경계인 띄어쓰기는 출력하지 않고, 형태소 시퀀스만 출력하게 된다. 입력과 마찬가지로 한글 외의 문자는 각 문자를 음절과 동등하게 처리하여 학습하였다.

“공이 데굴데굴 굴렀다”라는 문장이 입력되었을 때의 입출력 예시를 에 나타냈다.

표 2 본 모델의 입출력 예시

입력 문장	공이 데굴데굴 굴렀다.
인코더 입력	공 이 <s> 데 굴 데 굴 <s> 굴 렀 다 <s> .
디코더 출력	공 <NNG> 이 <JKS> 데 굴 데 굴 <MAG> 구 르 <VV> 었 <EP> 다 <EF> . <SF>
분석 결과	공/NNG 이/JKS 데굴데굴/MAG 구르/VV 었/EP 다/EF ./SF

또한 [3], [17] 등의 시퀀스 투 시퀀스를 사용하는 기존 논문이 단순히 음절 임베딩만을 인코더로의 입력에 사용한 것과 달리, 더 많은 정보를 입력하기 위해 입력 문장을 임베딩할 때, 음절 임베딩 외에도 음절 Bigram과 자소를 같이 임베딩하여 사용하였다. 본 모델의 추가적인 입력 요소를 포함한 입력 임베딩의 구조는 그림 3와 같다. 3.2.2와 3.2.3에서 각 입력 요소에 대해 설명한다.

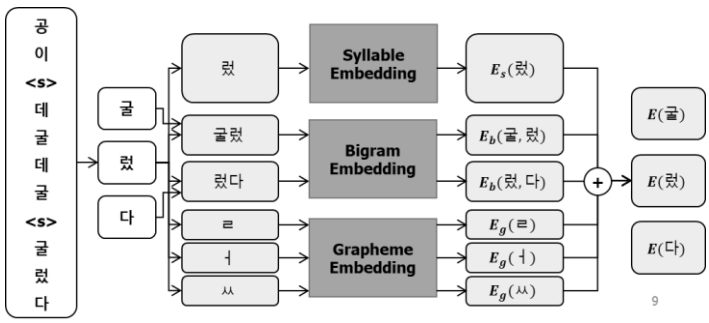

그림 3 본 모델의 문장 입력 임베딩 구조

3.2.2 음절 bigram 임베딩

음절 임베딩에 더불어, 음절 bigram 임베딩을 추가로 입력에 사용하였다. 음절 bigram 임베딩은 [2], [13] 등에서 형태소 분석에 추가 요소로 활용한 바 있다. 세종 말뭉치의 형태소 분석 데이터셋의 형태소는 90% 이상이 1음절 또는 2음절로 된 형태소이다. 따라서 만약 음절 unigram과 음절 bigram을 같이 사용한다면 입력 임베딩 자체에 대부분의 한국어 형태소 정보를 담을 수 있는 것이다. 따라서 음절 bigram 임베딩을 사용함으로써 음절 임베딩만으로는 부족한 형태소 형성 및 구분에 관한 정보를 더욱 풍부하게 학습할 수 있다.

본 모델에서는 그 음절의 직전 음절과의 bigram, 그 음절의 다음 음절과의 bigram 두 가지를 모두 임베딩하였다. 그리고 임베딩된 벡터는 음절 임베딩 벡터와 결합(concatenate)하여 사용하였다. 한글이 아닌 문자의 경우 영어, 숫자, 한자, 그 외로 분류하여 단순화하여 bigram이 너무 많아지지 않도록 하였다. 또한 문장의 처음과 끝에는 <SOS>, <EOS> 토큰을 추가하여 부족한 글자 수를 보충하였다.

3.2.3 자소 임베딩

본 논문에서는 음절을 구성하는 각 자소(Grapheme)의 임베딩을 추가로 학습하여 사용하였다. [1]에서 음절 임베딩 없이 자소 임베딩을 통한 형태소 분석기를 연구한 바 있다. 자소를 입력 단위로 사용하면 음절 단위 데이터에 자주 나타나는 오타 및 음절 변형을 통한 신조어에 효과적일 것이라 판단하였다.

본 모델에서는 인코더에 입력으로 들어가는 한글 음절을 초성, 중성, 종성으로 분해하여 각각 자소 임베딩 행렬을 사용하여 고정 차원의 벡터로 만들었다. 그리고 이 임베딩 벡터를 기존 음절 임베딩에 결합하여 사용하였다. 한글이 아닌 문자의 경우 영어, 숫자, 한자, 그

외로 분류하여 단순화하였다.

3.3 주의 기반 인코더-디코더

형태소 분석을 실제로 수행하는 신경망으로는 시퀀스 투 시퀀스, 그 중에서도 주의 기반 인코더-디코더 모델을 사용하였다. 주의 기반 인코더-디코더는 임베딩된 문장을 LSTM과 같은 순환 신경망을 통해 인코딩하고, 이 인코딩된 고정 크기의 벡터와 인코더의 출력 시퀀스를 이용하여 원형 복원된 형태의 형태소를 음절 단위로 연속적으로 출력한다. 본 모델에서는 인코더와 디코더의 순환 신경망을 구성하는 기본 뉴런으로 LSTM을 두 층으로 쌓아서 사용하였다. 인코더-디코더 부분의 구조는 그림 4와 같다.

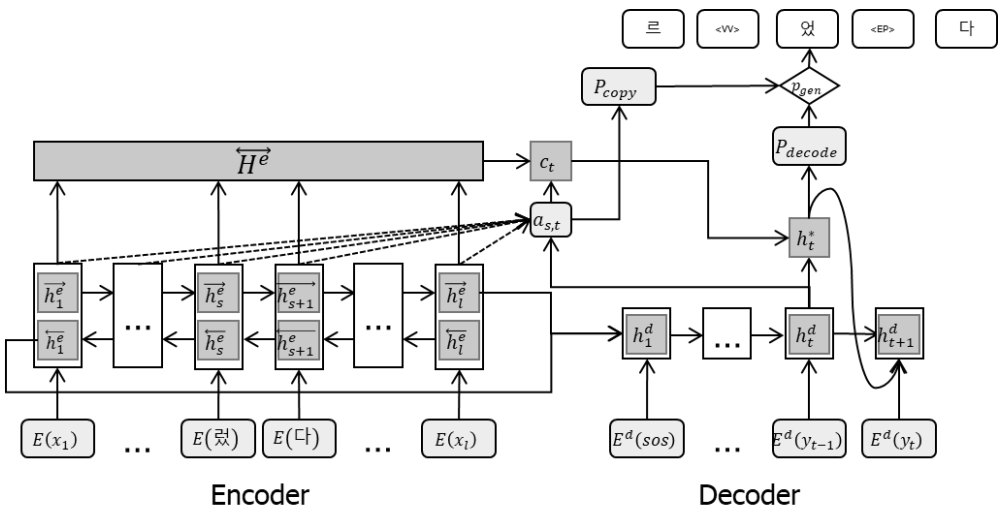

그림 4 본 모델의 인코더-디코더 구조

인코더는 양방향 구조를 적용하여, 문장 처음에서 시작하여 순차적으로 문장을 읽어들이는 LSTM과 문장 끝에서 시작하여 역순으로

문장을 읽어들이는 LSTM을 같이 사용하여 출력을 결합하였다. 문장의 s번째 입력에 대한 인코더 계산식은 아래와 같다.

$$\overrightarrow{h_s^e} = \text{LSTMs}\left(E(x_s), \overrightarrow{h_{s-1}^e}\right)$$
$$\overleftarrow{h_s^e} = \text{LSTMs}\left(E(x_s), \overleftarrow{h_{s+1}^e}\right)$$
$$\overleftrightarrow{h_s^e} = [\overleftarrow{h_s^e}, \overrightarrow{h_s^e}]$$

디코더는 입력 추가 구조(input-feeding)가 적용된 주의 기반 디코더[6]을 기반으로 하였다. 주의 기반 모델은 순환 신경망의 출력과 인코더의 출력의 관계를 계산, 주의 분포(attention distribution) a_t를 이용하는 모델이다. 이를 통해 음절 단위 입력으로 인해 문장이 길어지더라도 정확한 분석이 가능하다.

또한 미등록 음절 처리를 위해 Pointer-generator Network[11] 모델을 적용하였다. 모델인 Pointer-generator Network는 디코더 출력에 인코더의 입력 중 일부가 그대로 복사되어 출력될 확률을 고려하는 모델이다. 일반적인 주의 기반 디코더의 출력 확률 분포 $\mathrm{P}_{\mathbf{decode}}(y_t = w)$에 더불어, 입력의 주의 분포 $\mathrm{P}_{\mathbf{copy}}(y_t = w)$를 같이 고려함으로써 출력되는 결과를 보정한다. 예를 들어 학습 데이터셋에 나타나지 않아 기존 어휘에 없던 OOV(Out-of-Vocabulary)에 해당하는 음절 w가 입력으로 들어올 경우 기존 모델에서는 $\mathrm{P}_{\mathbf{decode}}(y_t = w)$가 0이 되어 디코더의 출력으로 나타날 수 없지만, Pointer-generator Network에서는 $\mathrm{P}_{\mathbf{copy}}(y_t = w)$가 존재하여 디코더의 출력으로 나타날 확률이 존재하게 된다. 따라서 Pointer-generator Network는 OOV가 많이 나타나는 인터넷 데이터에 적합하다. 또한 Pointer-generator Network는 [3]에서는 미등록 음절을 위해 적용한 CopyNet과 달리

copy 확률 p_{gen}을 학습하여 복사와 생성의 균형을 유지한다.

본 논문에서 구현한 디코더 구조의 상세한 수식은 아래와 같다.

$$h_t^d = \text{LSTMs}\left([E^d(y_{t-1}), h_{t-1}^*], h_{t-1}^d\right)$$

$$e_{s,t} = \overleftrightarrow{h_s^e} W_a h_t^d$$

$$a_{s,t} = \frac{\exp\left(e_{s,t}\right)}{\sum_{i=1}^{l} e_{i,t}} = \text{softmax}_s(\text{e}_{\text{s,t}})$$

$$c_t = \sum_{i=1}^{l} a_{i,t} \overleftrightarrow{h_i^e} = \overleftrightarrow{H^e} a_t$$

$$h_t^* = \tanh\left(W_{hc} c_t + W_{hd} h_t^d + b_h\right)$$

$$\text{P}_{\text{decode}}(y_t = w) = \text{softmax}_{\text{vocab}}(\text{W}_\text{o} h_t^* + b_o)$$

$$\text{P}_{\text{copy}}(y_t = w) = \sum_{i:w=x_i} a_{i,t}$$

$$\text{p}_{\text{gen}} = \sigma\left(w_{gc} c_t + w_{gh} h_t^* + b_g\right)$$

$$p(y_t = w) = p_{gen} P_{decode}(y_t = w) + \left(1 - p_{gen}\right) P_{copy}$$

제 4 장 실험

이 장에서는 본 논문에서 제안한 형태소 분석기의 성능을 평가한다. 1절에서는 실험에 사용한 데이터를 소개하고, 2절에서는 실험 환경을 설명한다. 3절에서는 본 논문에서 형태소 분석 성능을 평가하기 위한 방법을 소개하고, 4절에서는 실제 실험 결과를 통해 본 논문에서 제안한 모델의 형태소 분석 성능을 평가한다.

4.1 실험 데이터

4.1.1 세종 말뭉치

21세기 세종계획 말뭉치, 통칭 세종 말뭉치[④]는 국내 최대 형태소 분석 말뭉치이다. 세종 말뭉치에는 형태소 분석 및 품사 태깅이 된 한국어 문장이 총 1303218 문장이 있으며, 본 논문에서는 이 말뭉치를 Train:Validate:Test의 비율이 85:5:10가 되도록 임의로 나누어 세종 Train Set, Valid Set, Test set을 각각 만들어 사용하였다. 각 데이터셋에 대한 말뭉치에 대한 자세한 통계는 과 같다.

표 3 본 논문에서 사용한 세종 말뭉치 분할의 문장, 어절, 형태소 개수

count	Train	Valid	Test
문장	1107735	65160	130323
어절	12851369	753987	1514955
형태소	29213081	1714476	3444975

④ https://ithub.korean.go.kr/user/main.do

또한, 세종 말뭉치에서 공백을 제거한 데이터셋을 같이 사용하였다. 모델 학습 시 공백을 제거한 문장과 공백을 제거하지 않은 문장을 1:1 비율로 섞어서 사용하여 띄어쓰기 오류에 대한 강건함을 높였고, Test Set도 공백을 제거하여 4.3.2의 실험에서 사용하였다.

4.1.2 Korean Internet Morpheme Dataset

인터넷 텍스트 데이터에서의 형태소 분석 성능을 확인하기 위해 인터넷 텍스트 샘플을 만들었다. 실제 인터넷 텍스트를 수집하기 위해, 웹 크롤러를 통해 DCinside(www.dcinside.com), 클리앙(www.clien.com), mlbpark(www.mlbpark.com), youtube 댓글(www.youtube.com)에서 한국어 문장을 총 500개 수집하였다. 그리고 이 500개의 문장에 대해 인력으로 품사 태깅을 제외하고 형태소 원형 복원까지의 과정을 수행하였다. 단, 품사 태깅의 경우 모호성이 있을 수 있어 하지 않았다. 이를 통해 500쌍의 형태소 분석 데이터셋을 구축하여 형태소 원형 복원 성능을 평가할 수 있도록 하고, Korean Internet Morpheme Dataset이라고 하였다. 4.4.1과 4.4.4에서 이 데이터셋을 성능 평가에 활용하였다.

4.2 학습 환경 및 변수

각 네트워크에서 입출력 음절 임베딩은 100차원, 음절 bigram 임베딩은 50차원, 자소 임베딩은 10차원을 사용하였다. LSTM의 은닉 차원은 300차원으로 하였고, layer를 2개로 하였다. 학습 dropout값은 0.3으로 주었다. 학습 모델은 OpenNMT-py[4]를 기반으로 수정하여 사용하였다.

4.3 성능 평가 방법

성능 평가 기준으로는 일반적으로 형태소 분석 성능 평가에 사용하는 형태소 단위 F1-measure를 사용하였다. 또한, 품사 태깅 과정을 제외한 형태소 분할 및 원형 복원 과정만의 성능 역시 F1-measure를 통해 평가하였다. 형태소 분석 결과를 이용하여 구문 분석이나 의미 분석, 키워드 추출 등을 진행 시 형태소 분할 및 복원 과정에서의 오분석은 품사 오류와 달리 돌이킬 수 없기 때문에 이 성능을 추가로 확인하였다.

또한 형태소 분석기의 띄어쓰기 오류에 대한 강건함을 확인하기 위해, 띄어쓰기를 제거한 테스트 입력에 대한 형태소 분석 성능을 기존 입력에 대한 성능과 비교하였다.

그리고, 신조어에 대한 강인함을 확인하기 위한 실험 역시 하였다. 학습 데이터에 잘 등장하지 않거나 아예 등장하지 않는 형태소에 대한 분석 성능을 확인하기 위해 세종 Train set에서 분석 결과로 나타나는 형태소를 등장 빈도에 따라 나누어 분류하였다. 그리고 각 모델의 세종 Test set에서 형태소 분석 결과를 평가할 때, 형태소 빈도에 따라 각각 F1-measure를 측정하였다. 이와 관련된 통계는 에 정리하였다.

표 4 세종 Train set에서의 등장 빈도에 따른 형태소 종류 및 각 형태소 그룹의 세종 Test set에서의 등장 횟수

등장 빈도 in Train set	형태소 종류	등장 횟수 in Test set
OOV	-	15454
<= 0.00001%	159516	13993
<= 0.00010%	69816	63894
<= 0.00100%	18526	197030
<= 0.01000%	4839	484065
<= 0.10000%	803	705146
<= 1.00000%	76	804254
<=10.00000%	16	1161139

4.4 실험 내용 및 결과

4.4.1 형태소 분석 성능 비교

본 논문에서 제안한 모델의 성능 평가를 위한 실험으로, 형태소 분석 결과의 형태소 단위 F1-measure를 비교하는 실험을 하였다.

성능 평가를 위한 데이터셋으로는 4.1.1에서 설명한 세종 말뭉치에서의 test 데이터셋과 4.1.2에서 구축한 Korean Internet Morpheme Dataset을 이용하였다. 세종 말뭉치에 대해서는 품사를 제외하고 형태소의 원형 복원 형태만 비교하는 F1-measure와 품사 태깅까지 완료된 상태에서의 형태소 단위 F1-measure를 모두 확인하였다. Korean Internet Morpheme Dataset에서는 품사를 제외하고 형태소의 원형 복원 형태만을 비교하는 F1-measure만 비교하였다.

성능 비교 대상은 동일한 테스트셋에서의 성능을 확인하기 위해 직접 실행할 수 있도록 학습된 모델이 공개된 형태소 분석기와 비교하였다. konlpy[10]에서 제공하는 형태소 분석기 중 kkma와 komoran을 사용하고, khaiii와 [1]을 추가로 비교 대상 모델로 사용하였다. 또한, [3], [9], [13], [17], [19]의 논문에 나온 형태소 분석 성능을 같이 비교하였다. 이 논문들은 역시 세종 말뭉치에서의 형태소 단위 F1-measure를 형태소 분석 성능으로 제시하고 있으나, 본 논문과는 다른 데이터셋 분할을 사용하여 직접적인 비교는 힘들다.

논문에서 제안한 모델은 음절 임베딩에 추가된 요소인 음절 bigram 임베딩과 자소 임베딩의 성능을 확인하기 위해, 각 임베딩을 추가한 것과 추가하지 않은 모델을 비교하였다. 성능 비교 결과는 와 같다.

표 5 각 모델의 형태소 분석 성능 비교
단위: 형태소 단위 F1-measure
(*은 다른 테스트 셋 사용)

	모델 및 출처	주요 알고리즘	사전 사용 여부	품사 미고려		품사 포함
				세종 말뭉치	Korean Internet Morpheme Dataset	세종 말뭉치
기존 모델	이건일 et. al (2017)[17]	Seq2Seq	X	-	-	0.9715*
	Jung, Sangkeun et. al (2018)[3]	Seq2Seq	X	-	-	0.9708*
	이창기 (2013)[19]	S-SVM	O	-	-	0.9803*
	Na, Seung-Hoon, Kim Young-Kil (2018)[8]	CRF	O	-	-	0.9774*
	김선우, 최성필 (2018)[13]	BILSTM-CRF	O	-	-	**0.9877***
	kkma	HMM	O	0.8920	0.7979	0.8333
	komoran	HMM	O	0.9438	0.7990	0.8342
	khaiii	CNN	O	0.9544	0.7589	0.9421
	choi et. al (2016)[1]	BILSTM-CRF	X	0.9742	0.8008	0.9586
제안 모델	음절 임베딩	Seq2Seq	X	0.9814	0.8252	0.9723
	음절+자소 임베딩	Seq2Seq	X	0.9824	0.8321	0.9735
	음절+bigram 임베딩	Seq2Seq	X	0.9860	**0.8321**	0.9781
	음절+자소+bigram 임베딩	Seq2Seq	X	**0.9868**	0.8317	0.9793

우선 품사가 태깅된 세종 말뭉치에 대한 성능을 보면, 본 논문에서 제안한 세종 말뭉치의 성능이 기존 대부분의 연구의 성능보다 높음을 확인할 수 있다. 비록 세종 말뭉치에서 학습 데이터와 Test 데이터를 나누는 방식이 달라 직접적인 수치 비교는 힘들지만, [13]을 제외한 다른 형태소 분석기에 근접하거나 조금 더 높은 성능이 나타났다. 특히 음절 bigram 임베딩을 적용한 모델이 그렇지 않은 모델에 비해 약 0.6%가량 성능 향상이 있었다. 특히, 원형 복원에 사전을 사용하지 않는 모델 중에서는 가장 높은 성능이 나타났다. 자소 임베딩은 0.1% 성능 향상이 있었지만 음절 bigram 임베딩에 비해서는 효과가 크게 없었다.

한편, 품사가 태깅되지 않은 세종 말뭉치에 대한 성능 역시 기존에 공개되어있는 형태소 분석기에 비해 크게 높은 성능을 가진 것으로 나타났다. 또한 Korean Internet Morpheme Dataset에서의 형태소 분할 및 원형 복원 성능 역시 기존 형태소 분석기보다 높게 나타났다. 단, 이 데이터셋에 대해서는 자소 임베딩이 음절 bigram 임베딩과 근접한 성능을 나타냈으나, 두 임베딩을 같이 사용한 경우에 대해서는 오히려 성능이 약간 감소하였다.

4.4.2 띄어쓰기를 제거한 데이터에서의 성능 확인

형태소 분석기가 띄어쓰기에 대해 얼마나 강건한지 확인하기 위해, 띄어쓰기를 제거할 경우 형태소 분석의 정확도가 얼마나 감소하는지에 대한 실험을 하였다.

기존의 비교 가능한 형태소 분석기와 논문에서 제안한 모델에 대해서, 세종 test 데이터셋에서 공백을 제거한 입력에 대한 형태소 단위 F1-measure를 비교하여 공백을 제거하기 전에 비해 얼마나 성능이 감소하는 지 확인하여 공개되어있는 모델 및 띄어쓰기 모듈을 포함한 모델인 [13], [19]와 비교하였다. 실험 결과는 에 정리하였다.

본 논문에서 제안한 모델은 품사 포함, 품사 미부착 두 가지 경우 모두에 대해 띄어쓰기가 없는 데이터셋에서의 성능 감소가 0.01 전후로 나타났다. 특히 자소 임베딩과 음절 bigram 임베딩을 같이 사용하는 경우 띄어쓰기 제거에 따른 성능 감소가 최소로 나타났다. 반면 기존 형태소 분석기는 다수가 띄어쓰기가 없는 경우에 성능이 크게 감소하는 것을 확인할 수 있었다. 본 모델보다 성능이 높은 [13]은 본 논문에서 제안하는 모델과 달리 형태소 분할 및 원형 복원에서부터 사전 수록 여부를 입력 요소로 사용하는 사용하는 모델이다.

표 6 세종 말뭉치에서 공백을 제거한 입력에 대한 형태소 분석 성능 비교
단위는 형태소 단위 F1-measure
(*은 다른 테스트 셋 사용)

	모델 및 출처	주요 알고리즘	사전 사용 여부	Sejong 말뭉치 (품사 미고려)	Sejong 말뭉치 (품사 포함)
기존모델	이창기 (2013)[19]	S-SVM	O		0.9699 (-0.0104)*
	김선우, 최성필 (2018)[13]	BILSTM-CRF	O		0.9792 (-0.0085)*
	Kkma	HMM	O	0.8696 (-0.0224)	0.8141 (-0.0191)
	Komoran	HMM	O	0.8275 (-0.1163)	0.6841 (-0.1501)
	Khaiii	CNN	O	0.5491 (-0.4053)	0.4585 (-0.4836)
	Choi, et. al (2016)[1]	BILSTM-CRF	X	0.6941 (-0.2800)	0.6941 (-0.3025)
제안모델	음절 임베딩 (Baseline)	Seq2Seq	X	0.9681 (-0.0134)	0.9584 (-0.0139)
	음절+자소 임베딩	Seq2Seq	X	0.9691 (-0.0134)	0.9594 (-0.0140)
	음절+bigram 임베딩	Seq2Seq	X	0.9765 (-0.0095)	0.9677 (-0.0104)
	음절+자소+bigram 임베딩	Seq2Seq	X	0.9791 (-0.0077)	**0.9710 (-0.0083)**

4.4.3 형태소 등장 빈도에 따른 분석 성능 확인

형태소 분석기의 신조어 분석 능력을 확인하기 위해, 형태소의 등장 빈도별 분석 성능을 확인하는 실험을 하였다.

세종 말뭉치 중 Train set에서의 형태소를 등장 빈도에 따라 분류하여 세종 test set에서의 형태소 분석 F1-measure를 따로 측정하여 비교하였다. 그 결과 그래프를 그림 5 및 에 제시하였다.

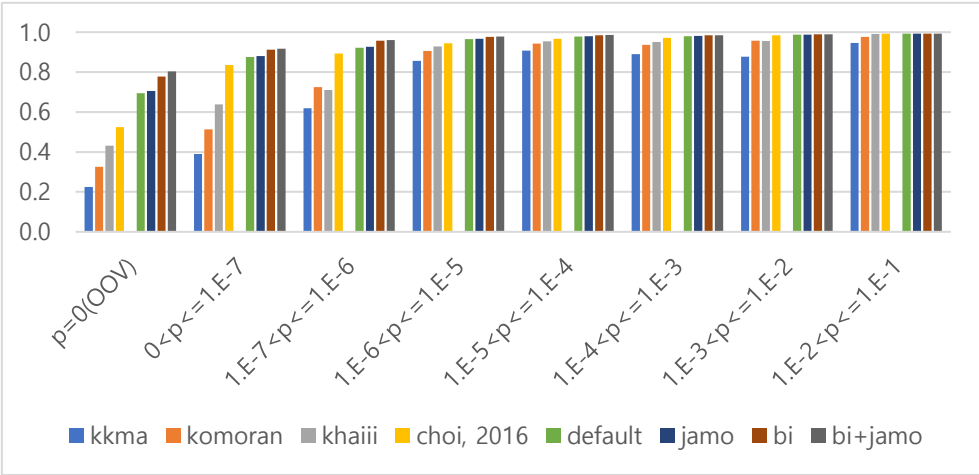

그림 5 형태소 등장 빈도에 따른 세종 테스트 셋에서의 분석 성능

표 7 OOV 및 등장 빈도가 낮은 형태소의 세종 테스트 셋에서의 분석 성능 표

		OOV	0<p<=1.E-7	1.E-7<p<=1.E-6
기존 모델	kkma	0.2245	0.3897	0.6193
	komoran	0.3258	0.5127	0.7253
	khaiii	0.4321	0.6377	0.7098
	Choi et al (2016)[1]	0.5242	0.8358	0.8938
제안 모델	음절 임베딩	0.6939	0.8755	0.9215
	음절+자소 임베딩	0.7058	0.8804	0.9262
	음절+bigram 임베딩	0.7775	0.9118	0.9565
	음절+자소+bigram 임베딩	**0.8037**	**0.9175**	**0.9598**

기존 공개 형태소 분석기에 비해 모든 경우에서 높은 성능을 나타냈고, OOV에 대해서도 0.7~0.8의 F1-measure가 나타났다. 반면 기존 공개 형태소 분석기는 0.2~0.5의 F1-measure가 나타났다.

전체적으로 등장 빈도가 낮은 형태소일수록 성능이 크게 차이나는

것을 확인할 수 있었고, OOV에서의 성능 차이가 가장 크게 나타나는 것을 확인할 수 있었다. 특히 음절 bigram을 적용한 모델이 그렇지 않은 모델에 비해 OOV에서의 F1-measure가 0.1 가까이 증가하는 것을 확인할 수 있었다.

4.4.4 OOV 한글 형태소의 Korean Internet Morpheme Dataset에서의 분석 성능

인터넷에서 직접 수집한 데이터셋인 Korean Internet Morpheme Dataset에서 세종 Train set에서 나타나지 않은 OOV 형태소 중 한글로 된 형태소를 골라 이에 대한 형태소 분석 성능을 확인하였다. 인터넷 데이터에는 형태소 분석이 애매한 한국어가 아닌 형태도 많이 나타났기 때문에 한글로 된 형태소에 대한 성능만 따로 확인하였다. 또한 이 실험에서는 F1-measure뿐만 아니라 precision과 recall을 같이 확인하였다. 그 결과를 **오류! 참조 원본을 찾을 수 없습니다.**에 정리하였다.

표 8 OOV 한글 형태소의 Korean Internet Morpheme Dataset에서의 분석 성능

	모델	precision	recall	F1-measure
기존 모델	kkma	0.4000	0.2261	0.2889
	komoran	0.3949	0.2191	0.2818
	khaiii	0.2820	0.4594	0.3495
	Choi et. al (2016)[1]	0.3280	0.5088	0.3989
제안 모델	음절 임베딩	0.4717	0.5300	0.4992
	음절+자소 임베딩	0.5113	**0.5618**	0.5354
	음절+bigram 임베딩	0.5983	0.4947	0.5416
	음절+자소+bigram 임베딩	**0.6026**	0.4982	**0.5455**

성능 확인 결과, 기존 모델에 비해 0.1 이상 높은 F1-measure가 나타났다. 특히 기존 모델에 비해 높은 것은 precision이었다. Precision이 높다는 것은 false positive가 적다는 것으로, 오분석으로 실제로 존재하지 않는 형태소를 생성하는 경우가 적다는 것을 의미한다. Recall의 경우에는 bigram 임베딩을 사용하는 것으로 인해 조금 낮아졌는데, 이는 bigram의 경우 기존 단어 형성 정보를 많이 담고 있기 때문에 기존 언어 상식과 다른 형태소가 새로 나타났을 경우 이를 제대로 분석해내지 못하는 경우가 있는 것으로 생각된다.

자소 임베딩의 경우 precision과 recall이 모두 상승하는 효과가 있었고, 특히 bigram 임베딩을 사용하지 않았을 때 더 큰 성능 향상이 있었다.

제 5 장 결론

본 논문에서는 한국어 인터넷 텍스트 데이터를 잘 분석하기 위한 한국어 형태소 분석기를 연구하였다. 인터넷 데이터의 특징인 띄어쓰기 문제와 OOV 문제를 해결하기 위해 시퀀스 투 시퀀스를 이용하고, 음절 bigram정보와 자소 정보를 같이 사용하여 형태소 분석을 하였다.

성능 평가 결과 시퀀스 투 시퀀스를 이용하는 것을 통해 사전이나 복잡한 전처리 없이도 충분히 경쟁력있는 분석 정확도가 나타났고, 음절 bigram 정보와 자소 정보가 형태소 분석 정확도를 높이는 것 역시 확인할 수 있었다. 특히 음절 bigram 정보를 통한 성능 향상이 뚜렷하게 나타났다.

본 모델은 사전 정보 및 기존 언어 지식을 최소한으로 사용하면서도 다른 모델과의 경쟁력이 있는 성능을 나타내는 모델으로서의 의미가 있다. 또한 정제된 데이터가 아니라 어절 구분이 제대로 안 되어있거나 새로운 단어가 많이 출연하는 데이터에서도 충분한 성능을 가지기 때문에 활용도가 높을 것이다.

앞으로 이 논문을 바탕으로 음절 bigram 임베딩이나 자소 임베딩을 단순 결합이 아닌 더 효율적으로 적용하는 연구나 인터넷 데이터의 초성체나 오타 등 다른 특징을 더 잘 고려하는 형태소 분석기 연구가 가능할 것으로 보인다.

참고 문헌

[1] Choi, Jihun, Jonghem Youn, and Sang-goo Lee. "A grapheme-level approach for constructing a Korean morphological analyzer without linguistic knowledge." *2016 IEEE International Conference on Big Data (Big Data)*. IEEE (2016)

[2] 池田大志, 進藤裕之, and 松本裕治. "辞書情報と単語分散表現を組み込んだリカレントニューラルネットワークによる日本語単語分割." *NLP* (2017): 879-882.

[3] Jung, Sangkeun, Changki Lee, and Hyunsun Hwang. "End-to-End Korean Part-of-Speech Tagging Using Copying Mechanism." *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 17.3 (2018): 19.

[4] Klein, Guillaume, et al. "Opennmt: Open-source toolkit for neural machine translation." *arXiv preprint arXiv:1701.02810* (2017).

[5] Lee, Sang-zoo, Jun-ichi Tsujii, and Hae-Chang Rim. "Hidden Markov model-based Korean part-of-speech tagging considering high agglutinativity, word-spacing, and lexical correlativity." *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics* (2000).

[6] Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

[7] Matteson, Andrew, et al. "Rich Character-Level Information for Korean Morphological Analysis and Part-of-Speech Tagging." *arXiv preprint arXiv:1806.10771* (2018).

[8] Na, Seung-Hoon. "Conditional random fields for korean morpheme segmentation and pos tagging." *ACM Transactions on Asian and Low-Resource Language Information Processing* 14.3 (2015): 10.

[9] Na, Seung-Hoon, and Young-Kil Kim. "Phrase-based statistical model for korean morpheme segmentation and POS tagging." *IEICE Transactions on Information and Systems* 101.2 (2018): 512-522.

[10] Park, Eunjeong L., and Sungzoon Cho. "KoNLPy: Korean natural language processing in Python." *Proceedings of the 26th Annual Conference on Human & Cognitive Language Technology.* Vol. 6. 2014.

[11] See, Abigail, Peter J. Liu, and Christopher D. Manning. "Get to the point: Summarization with pointer-generator networks." *arXiv preprint arXiv:1704.04368* (2017).

[12] Sutskever et al. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. (2014): 3104-3112.

[13] 김선우, and 최성필. "Bidirectional LSTM-CRF 기반의 음절 단위 한국어 품사 태깅 및 띄어쓰기 통합 모델 연구." *정보과학회논문지* 45.8 (2018): 792-800.

[14] 신준철, and 옥철영. "기분석 부분 어절 사전을 활용한 한국어 형태소 분석기." *정보과학회논문지: 소프트웨어 및 응용* 39.5 (2012): 415-424.

[15] 심광섭, and 양재형. "인접 조건 검사에 의한 초고속 한국어 형태소 분석." *정보과학회논문지: 소프트웨어 및 응용* 31.1 (2004): 89-99.

[16] 심광섭. "음절 단위의 한국어 품사 태깅에서 원형 복원." *정보과학회논문지: 소프트웨어 및 응용* 40.3 (2013): 182-189.

[17] 이건일, 이의현, and 이종혁. "Sequence-to-sequence 기반 한국어 형태소 분석 및 품사 태깅." *정보과학회논문지* 44.1 (2017): 57-62.

[18] 이상주, 임희석, and 임해창. "은닉 마르코프 모델을 이용한 두단계 한국어 품사 태깅." *한국정보과학회 언어공학연구회 학술발표논문집* (1994): 305-312.

[19] 이창기. "Structural SVM 을 이용한 한국어 띄어쓰기 및 품사 태깅 결합 모델." *정보과학회논문지: 소프트웨어 및 응용* 40.12 (2013): 826-832.

[20] 이충희, et al. "기분석사전과 기계학습 방법을 결합한 음절 단위 한국어 품사 태깅." *정보과학회논문지* 43.3 (2016): 362-369.

Abstract

Sequence-to-sequence based Korean Morphological Analyzer for Neologism and Spacing Error

Choe Byoengseo
Department of Computer Science and Engineering
The Graduate School
Seoul National University

Recently, as the mount of Internet text data is increasing, the demand for natural language processing for the data, especially data from Korean internet communities is also increasing. However, morphological analysis is essensial for Korean natural language processing

In order to analyze Internet text data, it is necessary to accurately perform morphological analysis even in a sentence with a spacing error, and enough original form restoration performance for an out-of-vocabulary input. However, existing Korean morphology analyzer often use dictionaries and complicate preprocessing for the restoration.

Based on the sequence-to-sequence model, we proposed a Korean morphological analyzer model that can effectively handle the spacing problem and OOV problem. In addition, the model proposed in this paper uses syllable bigram and grapheme as additional input

features. Our model don't use dictionary and minimizes rule-based preprocessing.

As a result, our best model achieves a 0.9793 morpheme F1-measure for Sejong corpus, which is superior to other morphological analyzers without dictionary. We also found that there was a performance reduction of around 1% for datasets without space. Our model also had high performance for OOV words and Internet sample dataset.

Keywords : Morphological Analysis, POS Tagging, Original form recovery, Sequence-to-sequence, Internet Text Data
Student Number : 2017-25969